DW_addsub_dx

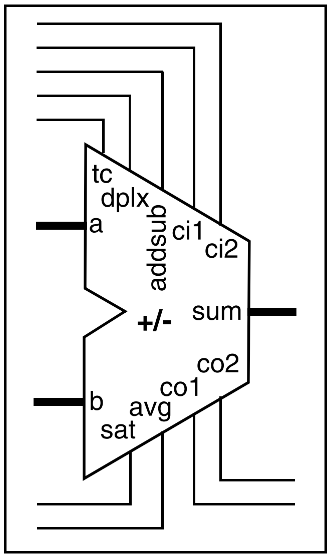
Duplex Adder/Subtractor with Saturation and Rounding

Version, STAR and Download Information: IP Directory

Features and Benefits

Revision History

- Selectable single full-width Add/Sub (simplex) or two smaller width Add/Sub operations (duplex)
- Selectable saturation mode
- Selectable average mode
- Selectable number system (unsigned or twos complement)
- Parameterized full word width
- Parameterized partial word width (allowing for asymmetric partial width operations)
- Carry-out signals (one for lower half and one for full and upper half) that numerically extend the calculated sum (maintaining full precision)
- Carry-in signals (one for full and lower half and one for upper half)



Description

DW_addsub_dx performs addition and subtraction of operands `a` and `b` as either:

- A single sum of *width* bits, or
- Two sums (one of *p1_width* bits and one of [*width* - *p1_width*] bits).

The sum or difference passes through a saturation unit and then through an arithmetic shifter. The saturation unit and shifter are controlled through the input ports, `sat` and `avg`, respectively.

The two's complement select input signal, `tc`, indicates the processing of unsigned or signed values. When `tc` is low, unsigned values are processed; when `tc` is high, signed values are processed.

Table 1-1 Pin Description

Pin Name	Width	Direction	Function
a	*width* bits	Input	Input data
b	*width* bits	Input	Input data
ci1	1 bit	Input	Full or part1 carry input
ci2	1 bit	Input	Part2 carry input

Table 1-1 Pin Description (Continued)

Pin Name	Width	Direction	Function
addsub	1 bit	Input	Add/subtract select input ■ 0 = Performs add ■ 1 = Performs subtract
tc	1 bit	Input	Two's complement select (active high)
sat	1 bit	Input	Saturation mode select (active high)
avg	1 bit	Input	Average mode select (active high)
dplx	1 bit	Input	Duplex mode select (active high)
sum	*width* bits	Output	Output data
co1	1 bit	Output	Part1 carry output
co2	1 bit	Output	Full width or part2 carry output

Table 1-2 Parameter Description

Parameter	Values	Description
width	≥ 4	Word width of `a`, `b`, and `sum`
p1_width	2 to *width* – 2	Word width of part1 of duplex Add/Sub

Table 1-3 Synthesis Implementations [a]

Implementation Name	Function	License Feature Required
rpl	Ripple Carry Synthesis Model	DesignWare
rpcs	Ripple Carry Select Synthesis Model	DesignWare
csm	Conditional Sum Synthesis Model	DesignWare

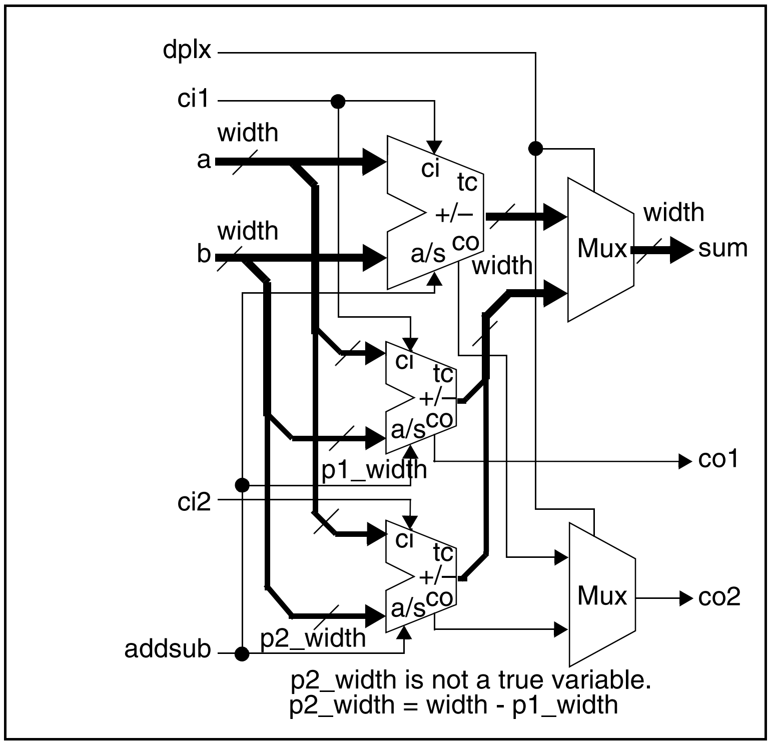
a. During synthesis, Design Compiler selects the appropriate architecture for your constraints. However, you may force Design Compiler to use one of the architectures described in this table.

Table 1-4 Simulation Models

Model	Function
DW01.DW_ADDSUB_DX_CFG_SIM	Design unit name for VHDL simulation
dw/dw01/src/DW_addsub_dx_sim.vhd	VHDL simulation model source code
dw/sim_ver/DW_addsub_dx.v	Verilog simulation model source code

Figure 1-1 shows a block diagram of DW_addsub_dx.

Figure 1-1 DW_addsub_dx Block Diagram



In addition to influencing saturation and shifting operations, `tc` influences the behavior of the carry output signals `co1` and `co2` such that, even for signed values, the carry outputs become a numeric extension of the sum or difference.

An example is given below:

```
Unsigned: "1111" + "0000" = "01111"
            15   +    0   =    15
          "1000" + "1000" = "10000"
             8   +    8   =    16
Signed:   "1111" + "0000" = "11111"
            -1   +    0   =    -1
          "1000" + "1000" = "10000"
            -8   +   -8   =   -16
```

Note that in the above example, the carry output (bold digit of binary result) does not always have the same value for signed versus unsigned operation.

The `addsub` input determines whether addition or subtraction is performed. When `addsub` is low, DW_addsub_dx performs an add; when `addsub` is high, subtraction is performed.

The `dplx` input selects whether a single full-width operation (`dplx` low) or two smaller width operations (`dplx` high) are performed.

Figure 1-2 shows a functional block diagram of DW_addsub_dx.

Figure 1-2 DW_addsub_dx Functional Block Diagram

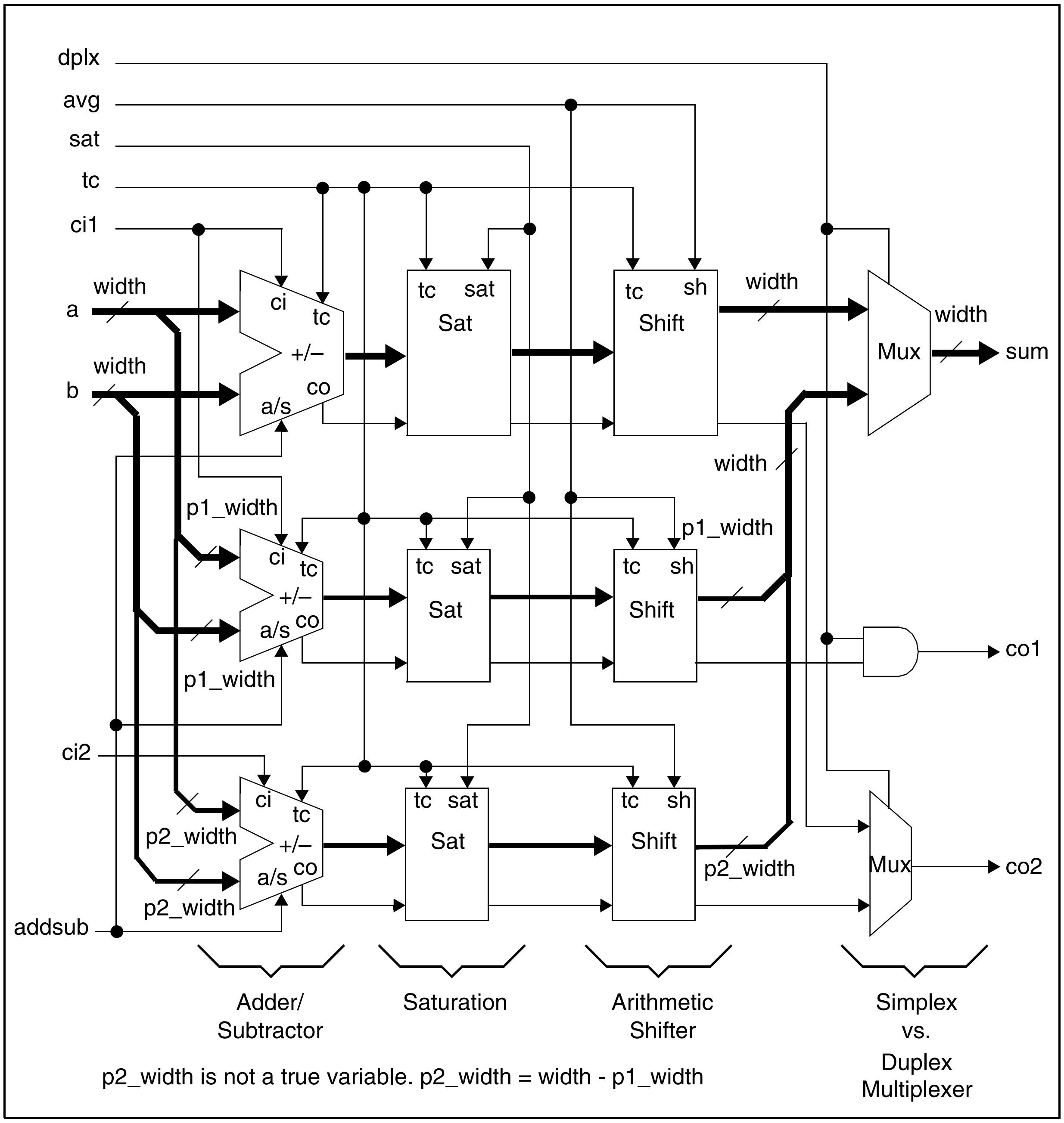


Table 1-5 Operating Modes

avg	sat	dplx	addsub	tc	Function
0	0	0	0	0	Simplex unsigned add operation: [co2,sum] = a + b + ci1
0	0	0	0	1	Simplex signed add operation: [co2,sum] = a + b + ci1
0	0	0	1	0	Simplex Unsigned subtract operation: [co2,sum] = a - b - ci1
0	0	0	1	1	Simplex signed subtract operation: [co2,sum] = a - b - ci1
0	0	1	0	0	Duplex unsigned add operation: [co1,p1_sum] = p1_a + p1_b + ci1 [co2,p2_sum] = p2_a + p2_b + ci2
0	0	1	0	1	Duplex signed add operation: [co1,p1_sum] = p1_a + p1_b + ci1 [co2,p2_sum] = p2_a + p2_b + ci2
0	0	1	1	0	Duplex unsigned subtract operation: [co1,p1_sum] = p1_a - p1_b - ci1 [co2,p2_sum] = p2_a - p2_b - ci2
0	0	1	1	1	Duplex signed subtract operation: [co1,p1_sum] = p1_a - p1_b - ci1 [co2,p2_sum] = p2_a - p2_b - ci2
0	1	0	0	0	Simplex saturated unsigned addition operation: [co2,sum] = unsigned_saturate(a + b + ci1)
0	1	0	0	1	Simplex saturated signed addition operation: [co2,sum] = signed_saturate(a + b + ci1)
0	1	0	1	0	Simplex saturated unsigned subtract operation: [co2,sum] = unsigned_saturate(a - b - ci1)
0	1	0	1	1	Simplex saturated signed subtract operation: [co2,sum] = signed_saturate(a - b - ci1)
0	1	1	0	0	Duplex saturated unsigned addition operation: [co1,p1_sum] = unsigned_saturate(p1_a + p1_b + ci1) [co2,p2_sum] = unsigned_saturate(p2_a + p2_b + ci2)
0	1	1	0	1	Duplex saturated signed addition operation: [co1,p1_sum] = signed_saturate(p1_a + p1_b + ci1) [co2,p2_sum] = signed_saturate(p2_a + p2_b + ci2)
0	1	1	1	0	Duplex saturated unsigned subtract operation: [co1,p1_sum] = unsigned_saturate(p1_a - p1_b - ci1) [co2,p2_sum] = unsigned_saturate(p2_a - p2_b - ci2)

Table 1-5 Operating Modes (Continued)

avg	sat	dplx	addsub	tc	Function
0	1	1	1	1	Duplex saturated signed subtract operation: [co1,p1_sum] = signed_saturate(p1_a - p1_b - ci1) [co2,p2_sum] = signed_saturate(p2_a - p2_b - ci2)
1	0	0	0	0	Simplex averaged unsigned addition operation: [co2,sum] = (a + b + ci1) >> 1
1	0	0	0	1	Simplex averaged signed addition operation: [co2,sum] = (a + b + ci1) >> 1
1	0	0	1	0	Simplex Averaged Unsigned Subtract operation: [co2,sum] = (a - b - ci1) >> 1
1	0	0	1	1	Simplex averaged signed subtract operation: [co2,sum] = (a - b - ci1) >> 1
1	0	1	0	0	Duplex averaged unsigned addition operation: [co1,p1_sum] = (p1_a + p1_b + ci1) >> 1 [co2,p2_sum] = (p2_a + p2_b + ci2) >> 1
1	0	1	0	1	Duplex averaged signed addition operation: [co1,p1_sum] = (p1_a + p1_b + ci1) >> 1 [co2,p2_sum] = (p2_a + p2_b + ci2) >> 1
1	0	1	1	0	Duplex averaged unsigned subtract operation: [co1,p1_sum] = (p1_a - p1_b - ci1) >> 1 [co2,p2_sum] = (p2_a - p2_b - ci2) >> 1
1	0	1	1	1	Duplex averaged signed subtract operation: [co1,p1_sum] = (p1_a - p1_b - ci1) >> 1 [co2,p2_sum] = (p2_a - p2_b - ci2) >> 1
1	1	0	0	0	Simplex averaged saturated unsigned addition operation: [co2,sum] = (unsigned_saturate(a + b + ci1)) >> 1
1	1	0	0	1	Simplex averaged saturated signed addition operation: [co2,sum] = (signed_saturate(a + b + ci1)) >> 1
1	1	0	1	0	Simplex averaged saturated unsigned subtract operation: [co2,sum] = (unsigned_saturate(a - b - ci1)) >> 1
1	1	0	1	1	Simplex averaged saturated signed subtract operation: [co2,sum] = (signed_saturate(a - b - ci1)) >> 1
1	1	1	0	0	Duplex averaged saturated unsigned addition operation: [co1,p1_sum] = (unsigned_saturate(p1_a + p1_b + ci1)) >> 1 [co2,p2_sum] = (unsigned_saturate(p2_a + p2_b + ci2)) >> 1

Table 1-5 Operating Modes (Continued)

avg	sat	dplx	addsub	tc	Function
1	1	1	0	1	Duplex averaged saturated signed addition operation: [co1,p1_sum] = (signed_saturate(p1_a + p1_b + ci1)) >> 1 [co2,p2_sum] = (signed_saturate(p2_a + p2_b + ci2)) >> 1
1	1	1	1	0	Duplex averaged saturated unsigned subtract operation: [co1,p1_sum] = (unsigned_saturate(p1_a - p1_b - ci1)) >> 1 [co2,p2_sum] = (unsigned_saturate(p2_a - p2_b - ci2)) >> 1
1	1	1	1	1	Duplex averaged saturated signed subtract operation: [co1,p1_sum] = (signed_saturate(p1_a - p1_b - ci1)) >> 1 [co2,p2_sum] = (signed_saturate(p2_a - p2_b - ci2)) >> 1

Related Topics

- Math – Arithmetic Overview
- DesignWare Building Block IP User Guide

HDL Usage Through Component Instantiation - VHDL

```
library IEEE,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DW_Foundation_comp_arith.all;

entity DW_addsub_dx_inst is
  generic ( inst_width    : NATURAL := 24;
            inst_p1_width : NATURAL := 16 );
  port ( inst_a      : in std_logic_vector(inst_width-1 downto 0);
         inst_b      : in std_logic_vector(inst_width-1 downto 0);
         inst_ci1    : in std_logic;
         inst_ci2    : in std_logic;
         inst_addsub : in std_logic;
         inst_tc     : in std_logic;
         inst_sat    : in std_logic;
         inst_avg    : in std_logic;
         inst_dplx   : in std_logic;
         sum_inst    : out std_logic_vector(inst_width-1 downto 0);
         co1_inst    : out std_logic;
         co2_inst    : out std_logic  );
end DW_addsub_dx_inst;

architecture inst of DW_addsub_dx_inst is
begin

  -- Instance of DW_addsub_dx
  U1 : DW_addsub_dx
    generic map ( width => inst_width,  p1_width => inst_p1_width )
    port map ( a => inst_a, b => inst_b,
               ci1 => inst_ci1, ci2 => inst_ci2, addsub => inst_addsub,
               tc => inst_tc, sat => inst_sat, avg => inst_avg,
               dplx => inst_dplx, sum => sum_inst,
               co1 => co1_inst, co2 => co2_inst  );
end inst;

-- pragma translate_off
configuration DW_addsub_dx_inst_cfg_inst of DW_addsub_dx_inst is
  for inst
  end for; -- inst
end DW_addsub_dx_inst_cfg_inst;
-- pragma translate_on
```

HDL Usage Through Component Instantiation - Verilog

```
module DW_addsub_dx_inst( inst_a, inst_b, inst_ci1, inst_ci2, inst_addsub,
                          inst_tc, inst_sat, inst_avg, inst_dplx, sum_inst,
                          co1_inst, co2_inst );

  parameter width = 24;
  parameter p1_width = 8;

  input [width-1 : 0] inst_a;
  input [width-1 : 0] inst_b;
  input inst_ci1;
  input inst_ci2;
  input inst_addsub;
  input inst_tc;
  input inst_sat;
  input inst_avg;
  input inst_dplx;
  output [width-1 : 0] sum_inst;
  output co1_inst;
  output co2_inst;

  // Instance of DW_addsub_dx
  DW_addsub_dx #(width, p1_width)
    U1 ( .a(inst_a), .b(inst_b), .ci1(inst_ci1), .ci2(inst_ci2),
         .addsub(inst_addsub), .tc(inst_tc), .sat(inst_sat),
         .avg(inst_avg), .dplx(inst_dplx),
         .sum(sum_inst), .co1(co1_inst), .co2(co2_inst) );

endmodule
```

Revision History

For notes about this release, see the *DesignWare Building Block IP Release Notes*.

For lists of both known and fixed issues for this component, refer to the STAR report.

For a version of this datasheet with visible change bars, click here.

Date	Release	Updates
March 2019	DWBB_201903.0	■ Added this Revision History table and the document links on this page